



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/449,782	11/26/1999	JAMES MCKEETH	MICS:0194	6698
52142 7590 10/17/2007 FLETCHER YODER (MICRON TECHNOLOGY, INC.) P.O. BOX 692289 HOUSTON, TX 77269-2289			EXAMINER STEELMAN, MARY J	
			ART UNIT	PAPER NUMBER
			2191	
			MAIL DATE	DELIVERY MODE
			10/17/2007	PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

**Office Action Summary**

Application No.

09/449,782

Applicant(s)

MCKEETH, JAMES

Examiner

MARY STEELMAN

Art Unit

2191

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☒ Responsive to communication(s) filed on 26 July 2007.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 1-21 and 23-25 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-21 23-25 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

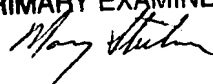
- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**MARY STEELMAN  
PRIMARY EXAMINER**



**Attachment(s)**

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO/SB/08)  
Paper No(s)/Mail Date \_\_\_\_\_
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: \_\_\_\_\_

Art Unit: 2191

### **DETAILED ACTION**

1. This Office Action is in response to Decision from Pre-Appeal Brief Review notice mailed 08/13/2007. Claims 1-21 and 23-25 are pending. Prosecution has been reopened.

#### ***Response to Arguments***

2. Applicant's arguments with respect to claims have been considered but are moot in view of the new ground(s) of rejection.

#### ***Claim Rejections - 35 USC § 103***

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

4. Claims 1-21 and 23-25 are rejected under 35 U.S.C. 103(a) as being unpatentable over USPN 6,182,279 to Buxton, in view of USPN 5,758,154 to Qureshi, in view of "The Windows NT Command Shell" by Tim Hill (hereinafter Hill) (1998), and further in view of Applicant's Admitted Prior Art (hereinafter AAPA).

Per claim 1:

-invoking, by an application, a call of a command line utility, the application providing an identifier in the call of the command line utility, wherein the command line utility is a utility executable from a command line prompt;

-receiving output from the command line utility;

Art Unit: 2191

- storing the command line utility output in a system storage at a location identified by the identifier;
- retrieving, by the application, the command line utility output from the system storage at the location identified by the identifier.

Buxton disclosed invoking a utility (col. 8, line 7) to modify and store the customized component/template component in the registry, using a (col. 13, lines 8-15) template Storage DLL to manage registration of templated components to keys, subkeys, etc. Col. 8, lines 8-11, modify registry Col. 14, lines 20-28, store at a registry key (a location identified by an identifier) Col. 10, lines 8-10, OLE libraries use the registry key information to find information about the OLE control (key values) (retrieving, by the application, the command line utility output from the system storage at the location (key / subkey location) identified by the identifier (the key). Buxton disclosed (col. 8, lines 45-50) a user interface or command line interpreter is used to invoke the application (the template customization application) that ultimately stores customization information as output in the registry.

Qureshi more clearly disclosed (col. 3, lines 26-46) an application call to a registration routine via system calls, to store an identifier in the registry. Col. 4, lines 2-7, invoke the registration routine, pass configuration file to the registration routine, open the configuration file and add the configuration file information to the registry. Col. 4, lines 58-67, DllRegisterServer routine Col. 5, lines 56-59, "The application programs invoke operating system routine in order to access the various services provided by the operating system...The application programs invoke the

Art Unit: 2191

registration...provided by the registration DLL to register...configuration information...The registration...routines of the registration DLL call operating system routines to write information to the registry file...” Col. 8, lines 48-67, “DllRegisterServer opens the specified key...calling the system routine RegOpenKey...DllRegisterServer creates the key...”

More explicitly,

Hill disclosed WindowsNT shell commands, CMD redirection, including the pipe redirection.

(page 2) “By far the most common use for a console window is to execute the Windows NT command shell. The command shell defines the Windows NT scripting language and is responsible for processing scripts, as well as commands typed at the keyboard.”

“The command shell is a console application.”

(page 4)

#### **Table 2.1 Special Characters in a Shell Prompt**

Character	Description
\$B	Pipe ( ) character.

#### **(page 6) CMD.EXE and COMMAND.COM**

Because all commands are actually executed by CMD.EXE (the Windows NT command shell)...

#### **(page 9) Command Redirection**

Art Unit: 2191

Most console applications and **commands generate output**, and many accept input. This input or **output is in the form of a stream of characters** (either ANSI or Unicode). Applications generally work with up to three streams, as follows:

**The command shell provides facilities to change the default stream input and output. These facilities are accessed by placing special command redirection symbols in a command.**

**(page 10) Table 2.4 Command Redirection Symbols**

Symbol	Description
>file	<b>Redirects command output to the file specified.</b> You can also use a standard device name such as LPT1, CON, PRN or CONOUT\$ as the file name. Any preexisting contents of the file are lost.
cmd1   cmd2	<b>Pipes the command output of cmd1 to the command input of cmd2.</b> Multiple pipe characters are allowed, creating a chain of commands, each sending output to the next command in the chain.

Command redirection symbols are not visible to the command. The shell processes them before the command is executed and they are not passed as arguments to the command. The <, >, and | symbols are reserved shell characters. If these symbols must be passed as command arguments, instead of being used as redirection symbols, then they must be escaped using the ^ character.

The > redirection symbol redirects command output to the specified file.

Art Unit: 2191

**The | (pipe) redirection symbol sends the command output of cmd1 to the command input of cmd2. For example:**

**C:\>dir | sort**

This example sends the command output of the DIR command to the command input of the SORT command. The output from the SORT command is then displayed. Alternatively, the SORT output can be sent to another command. For example:

**C:\>dir | sort | more**

This example sends the command output of the DIR command to the command input of the SORT command. Then, the command output of the SORT command is sent to the command input of the MORE command. Finally, the command output of the MORE command is displayed.

**(page 12) When the command shell processes a pipe (|) symbol, it actually runs both commands specified simultaneously. The right hand command is suspended until the left hand command begins generating command output. Then, the left hand command wakes up and processes the output. When this output has been processed, the command is again suspended until more input is available. The synchronization of both commands is handled automatically by the command shell and Windows NT.**

**The pipe symbol is both a redirection symbol and a compound command symbol. Compound commands are discussed in the next section.**

**(page 12) Using Command Filters**

Art Unit: 2191

The previous sections showed how individual commands can be combined using command redirection and compound command symbols. Although command redirection can be used with any command, it is most effective with commands that are specifically designed as command filters. Generally, a command filter reads command input, permutes, or processes the input in some manner, and then writes the permuted input to its command output. **Command filters are typically connected to other commands and each other via the pipe (|) redirection symbol.**

Command filters are frequently used in scripts to extract specific information needed by a script. Typically, a FIND command can filter the output of a command, extracting only the line (or lines) which contain the required information. The script can then further process this filtered data. Many of the sample scripts in Part II use this technique.

Windows NT provides four command filters:

- The MORE command, which is used to paginate command output.
- The SORT command, which can sort command output alpha-numerically.
- The FIND command, which filters lines that contain a specified text string.
- The CLIP [RK] command, which captures command output to the Windows NT clipboard.

AAPA disclosed FIG. 1 (page 1, line 28), an application invoking the command line utility. AAPA disclosed (page 1, line 19) command line utility output, piped to a temporary file, said temporary file created by the piping operation. AAPA notes problems associated with such



Art Unit: 2191

a temporary file: the command line utility may not have file creation privileges on the computer system (page 1, line 29), an error may be generated if the disk is too full to create a new file (page 2, line 2-3), there may be a naming error (page 2, line 4), or the system may be disk-less, not providing a mechanism through which a user initiated file input / output is possible (page 2, line 5). Applicant's invention claims to overcome the need for temporary storage by sending output directly to a registry file location. Applicant claims novelty in providing a mechanism by which an application program may obtain output from a command line utility without the need to create a temporary file.

However, it would have been obvious, to one of ordinary skill level in the art, at the time of the invention, to modify the method / system / storage device, as disclosed in AAPA, FIG. 1, to overcome the possible disadvantages noted above, as related to creating a temporary file, using WindowsNT known redirection and piping commands, because piping the output of a script command, to be used as an input to a registry edit command is an efficient use of resources. The results are as expected. Registry parameter values in a registry database storage location are modified without the need to create a temporary storage location. One would be motivated to ensure that sufficient resources exist to store utility output, and to otherwise manage memory.

The combination is obvious because teachings are found in the prior art, and combined, with no change in functionality. It is a mere use of common sense by one skilled in the art to select and combine such known elements with no new function, i.e., a predictable result.

Art Unit: 2191

The predictable result, utility output directly stored to a system storage, at a location identified by an identifier. A subsequently invoked application will retrieve the modified values from the system registry.

**Regarding claim 2, Buxton teaches:**

-providing the identifier comprises providing an identifier that identifies one or more entries in a system registry database.

(Fig. 2, item 205 and col. 13, lines 14-15, "...registry keys are created..." Also see col. 14, lines 29-59, "To facilitate loading of template onto another system...a number of registration key or subkey are included with template. Each template may have the keys 450A-I, as illustrated in Fig. 4C...Key 450H contains information indicating the name of the storage object in template storage file where initialization data...may be located...Key 450I contains information identifying the CLSID...)

**Regarding claim 3, Buxton teaches:**

-providing a root key identifier.

(Col. 11, line 2: "Most OLE object application information is stored in subkeys under the CSLID root key..." Also see col. 17, lines 35-41, "Component loader loads, verifies and checks the license of a component by replacing in registry the InProcessServer 32 entry, i.e. key 450A...and adding additional registry keys 450B-J, as previously described, that will let the component loader (receiving a root key identifier) then load the correct OLE control.")

**Regarding claim 4, Buxton teaches:**

-providing a sub-key identifier.

(Col. 11, line 2 and col. 14, line 31: To facilitate loading of template...a number of registration or subkey are included with template...")

**Regarding claim 5, Buxton teaches:**

-system registry database comprises an operating system registry database.

(Col. 4, line 49: "Operation of computer system is generally controlled...by operating system software, such as...Windows95...")

**Regarding claim 6, Buxton teaches:**

-providing a system storage identifier.

(Col. 12, lines 20-21, "...users identify...templates to be packaged..." Also for another example of receiving a system storage identifier, see col. 20, lines 42-45, "...relevant character string from the registry is converted to CLSID. The component loader (receives a system storage identifier) then calls the GetClassObject to retrieve the real component's class factory...")

**Regarding claim 7, Buxton teaches:**

-providing the system storage identifier comprises providing an identifier indicating a system registry.

Art Unit: 2191

(Col. 10, line 66 – col. 11, line 4: A CLSID identifies the functionality of an object class that can display...access to property values...A subkey is used by an OLE to find out information about the control.”)

**Regarding claim 8, Buxton teaches:**

-providing an identifier indicating shared system memory.

(Col. 8, lines 6-7: “OLE libraries (shared) comprise the set of system-level services in accordance with the OLE specification...”)

**Regarding claim 9, Buxton teaches:**

-providing the identifier indicating shared system memory identifies a system clipboard memory.

(Col. 11, line 6: “An FORMATETC...is an OLE data structure which acts in a generalized clipboard format...”)

**Regarding claims 10, Buxton teaches:**

-receiving output directly from the command line output utility.

As an example, a utility modifies (utility output) the registry (col. 8, lines 8-11).

**Regarding claim 11, Buxton teaches:**

-receiving output from the command line output utility through a subsequent command line output routine.

Art Unit: 2191

As an example, (col. 8, lines 28-29) "Data items within the registry are retrievable (receive output) via calls (from utility call) to the WIN32 APIs."

**Regarding claim 12, Buxton teaches:**

-associating each line of command line utility output with a line identifier in the system storage.

As an example, (col. 3, lines 1-9) "Template storage with a means for indexing, including key information associated with the template. "...a memory having one or more locations, means for indexing one or more locations within the memory..." Also col. 13, lines 35-44, templates are stored with an enumerated decimal number: "Each template is stored in an ISTORE whose name is unique...and may have the form TEMPLEnnn, where nnn may be a decimal number.")

**Regarding claim 13, Buxton teaches:**

-setting each line identifier to a value corresponding to a position of that line in the command line utility output.

(Rejection of claim 12 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 13 is rejected under the same rationale as claim 12.)

**Regarding claim 14, Buxton teaches:**

-setting a default value of the provided identifier to equal the total number of command utility output lines stored in the system storage. (Rejection of claim 12 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 14 is rejected under the same rationale as claim 12.)

**Regarding claim 15, Buxton teaches:**

A program storage device, readable by a computer, comprising instructions stored on the program storage device for causing the computer to:

- cause an application to invoke a call of a command line utility, the application providing an identifier in the call of the command utility;
- receive output from the command line utility;
- store the command line utility output in system storage at a location identified by the identifier;
- cause the application to retrieve the command line utility output from the storage at the location identified by the identifier.

See rejection of limitations in claim 1 above. This is a “program storage device” version of claim 1. See Figure 2 regarding Buxton’s disclosure of a “program storage device.”

**Regarding claim 16, Buxton teaches:**

- instructions to store command line utility output in an operating system registry database.

As an example (Fig. 2, item 205 and col. 13, lines 14-15), “...registry keys are created...” and (col. 13, lines 10 – 15) “...Template storage DLL ensures all additional registry keys...are created...” Modified components cause the registry keys to be created / edited / modified (REGEDIT utility).

**Regarding claim 17, Buxton teaches:**

Art Unit: 2191

-instructions to store command line utility output in an operating system maintained volatile memory.

As an example, (Fig. 1, item 110-volatile storage).

**Regarding claim 18, Buxton teaches:**

-instructions to receive one or more lines of output from the command line utility.

See rejection of limitation in claim 1 above.

-instructions to store each of said one or more lines of output in the system storage.

As an example, (col. 14, lines 26-29) "The remainder of the operating system registry entries are generated by code (instructions to store) in the template storage DLL and are stored in both registry (store output / modified component data in system storage) and the template.")

**Regarding claim 19, Buxton teaches:**

-instructions to associate a unique identifier with each of the one or more lines of output stored in the system storage.

See rejection of limitations in claim 2 above.

**Regarding claim 20, Buxton teaches:**

-instructions to set a value associated with the received identifier in the system storage equal to the number of lines of output stored in the system storage.

Art Unit: 2191

(Rejection of claim 18 is incorporated and further claim contains limitations as recited in claim 12. Therefore claim 20 is rejected under the same rational as claim 12.)

**Regarding claim 21, Buxton teaches:**

A computer system, comprising:

- a processor;
- a command line utility;
- an application executable on the processor, the application to call the command line utility, the application to provide an identifier in the call;
- a system storage having a location identified by the identifier, the location identified by the identifier to store an output of the command line utility,
- the application to retrieve the command line utility output from the location identified by the identifier.

As an example, see FIG. 1. Claim 21 contains limitations as recited in claim 1, therefore claim 21 is rejected under the same rational as claim 1.)

**Regarding claim 23, Buxton teaches:**

- the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility...
- wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.



Art Unit: 2191

Col. 8, lines 6-7 disclose the OLE libraries comprise the set of system level services (system utilities). As an example of system utilities (col. 20, lines 17-43) Buxton disclosed reading a sub-key from the registry, use the output to determine the real component CLSID, determine whether a valid certificate and license exist, pipe the relevant character string to a CLSID, etc. Additionally see rejection of claim 1 above.

**Regarding claim 24, Buxton teaches:**

-the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility...

-wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.

This is a 'program storage device' version of claim 23 above. See rejection of claim limitations in claims 15 and 23 above.

**Regarding claim 25, Buxton teaches:**

-the command line utility comprises a first command line utility, the system further comprising a second command line utility, the application to invoke a call that causes output of the second command line utility to be piped to the first command line utility...

-the location identified by the identifier to store output of the first command line utility.

This is a 'system' version of claim 23 above. See rejection of claim limitations in claims 21 and 23 above.

***Conclusion***

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

10/10/2007

**MARY STEELMAN  
PRIMARY EXAMINER**

